Modellierung von Zugriffsrichtlinien für offene Systeme

Wolfgang Dobmeier und Günther Pernul

Lehrstuhl für Wirtschaftsinformatik I
Universität Regensburg
D-93040 Regensburg
{wolfgang.dobmeier, guenther.pernul}@wiwi.uni-regensburg.de

Abstract: Für offene Systeme im Internet wie Web Services ist die Sicherheit der angebotenen Dienste von entscheidender Bedeutung. Ein wichtiger Baustein dabei ist die Umsetzung der Zugriffskontrolle. Dazu sind besondere Zugriffskontrollmodelle nötig, die über traditionelle Ansätze hinausgehen. Bisher fehlte es jedoch an einer Möglichkeit, Zugriffsrichtlinien für solche Modelle in leicht verständlicher Form spezifizieren zu können. Der Beitrag stellt neben einem passenden Zugriffskontrollmodell eine pragmatische Methode zur Modellierung von Zugriffsrichtlinien in Form eines UML-Profils vor. Die Methode wird an einem Beispielszenario illustriert.

1 Einleitung

Offene Systeme gewinnen in der modernen Zeit an Bedeutung. Beispiele für solche Systeme können allgemeine Anwendungen des eCommerce oder eGovernment-Portale sein. Eine Möglichkeit, offene Systeme technisch umzusetzen, sind Web Services. So könnte ein Unternehmen Dienste für jeden Benutzer offen nach außen hin anbieten, wie es z.B. Amazon[1] mit einem umfangreichen Spektrum, von der Abfrage von Preisen bis hin zur Abwicklung von privaten Buchangeboten, demonstriert.

Das bedeutet, dass diese Dienste auch geschäftskritische Bedeutung haben. Daher muss auf die Sicherheit dieser Dienste besonderes Augenmerk gelegt werden. Offene Systeme weisen zudem einige Charakteristika auf, die in herkömmlichen Systemen nicht anzutreffen sind. So werden z.B. anfragende Benutzer oftmals dem System nicht a priori bekannt sein, außerdem kann die Anzahl von Ressourcen und Benutzern sehr groß sein. Auf der einen Seite sollen also Ressourcen all denjenigen, die dafür berechtigt sind, zugänglich gemacht werden, auf der anderen Seite zugleich vor einem Zugriff durch Unberechtigte geschützt werden. Diese Aufgabe kommt der Zugriffskontrolle zu. Durch die Definition von Zugriffsrichtlinien (Policies) kann festgelegt werden, welche Subjekte auf welchen Objekten welche Operationen durchführen können. Dies war bereits in der Vergangenheit eine fehleranfällige und mühsame Tätigkeit.

Aus den o.g. Gründen muß die Zugriffskontrolle jedoch in offenen Systemen leistungsfähiger als in herkömmlichen Systemen gestaltet werden. Dies manifestiert sich zum Einen in

[1] `http://solutions.amazonwebservices.com`

der Verwendung neuartiger Modelle zur Zugriffskontrolle [DDCdVS05], die auf den Attributen von Subjekten und Objekten basieren (siehe Abschnitt 2). Zum Anderen hat sich unter dem Gesichtspunkt der Anwendungsarchitektur dabei eine logische Trennung der Bereiche Spezifikation der Zugriffsrichtlinien, Fällen der Zugriffskontrollentscheidung und Durchsetzen der Entscheidung herausgebildet. So definiert der offene Standard XACML [OAS05] neben einem XML-Dialekt zum Beschreiben von Zugriffsrichtlinien eine generische Architektur, die u.a. zwischen einem Policy Administration Point (PAP), Policy Decision Point (PDP) und Policy Enforcement Point (PEP) unterscheidet. Andere Autoren in der Literatur wie z.B. [YT05] folgen dieser Trennung in ähnlicher Weise.

Durch diese Separierung ist es also möglich, die Zugriffsrichtlinien getrennt von der Zugriffskontrolllogik zu verwalten, so dass dazu auch spezielle Werkzeuge mit eigenen Spezifikationsmöglichkeiten verwendet werden können. Jedoch mangelt es bisher daran. Durch die höhere Flexibilität (und Komplexität) attributbasierter Zugriffskontrollmodelle ist die Textform zur Dokumentation der Zugriffsrichtlinien nur eingeschränkt geeignet. So ist die XML-basierte Notation von XACML für den menschlichen Leser schwer zugänglich. Es gibt zwar vereinzelt Editoren für XACML (z.B. der UMU-XACML-Editor[2]), diese können jedoch nicht intuitiv bedient werden. Auch andere Möglichkeiten der Spezifikation von Richtlinien, wie z.B. graphbasierte [KMPP05] oder logikbasierte Ansätze [ZRG04], sind für ungeschulte Benutzer eher schwer verständlich. Da eine einfache und damit verständliche Definition der Zugriffsrichtlinien aber essentiell für die Sicherheit eines Systems ist, folgern wir, dass eine übersichtliche und gleichzeitig aber leistungsfähige Methode zur Richtlinien-Spezifikation notwendig ist. Im vorliegenden Beitrag präsentieren wir auf Basis bestimmter Anforderungen einen pragmatischen Ansatz zur grafischen Spezifikation von Zugriffsrichtlinien für offene Systeme. Er ist eine Weiterentwicklung des in [PDMP05] nur rudimentär dargelegten Vorgehens.

Im weiteren Verlauf des Beitrags wird in Abschnitt 2 ein auf die Anforderungen von offenen Systemen zugeschnittenes Zugriffskontrollmodell behandelt. Abschnitt 3 führt ein UML-Profil ein, mit dem Richtlinien für das vorgestellte Modell entworfen werden können; dies wird an einem Beispielszenario verdeutlicht. Der 4. Abschnitt geht auf verwandte Arbeiten ein. In Abschnitt 5 wird abschließend ein Ausblick auf zukünftige Entwicklungen gegeben.

2 Zugriffskontrolle für offene Systeme

Wie bereits in Abschnitt 1 erwähnt, weist das Problem der Zugriffskontrolle in offenen Systemen einige Besonderheiten auf. Traditionelle Zugriffskontrollmodelle wie die rollenbasierte Zugriffskontrolle (RBAC) [FSG+01], die sich am Aufbau von Organisationen orientiert, sind ungeeignet für den Einsatz an dieser Stelle. Gründe liegen u.a. darin, dass traditionelle Modelle identitätsbasiert sind, während in offenen Systemen die Identität von Subjekten zugunsten aller Eigenschaften der Subjekte in den Hintergrund tritt. Außerdem erschwert die potenziell große Zahl von Subjekten und Objekten das Management der

[2] `http://xacml.dif.um.es`

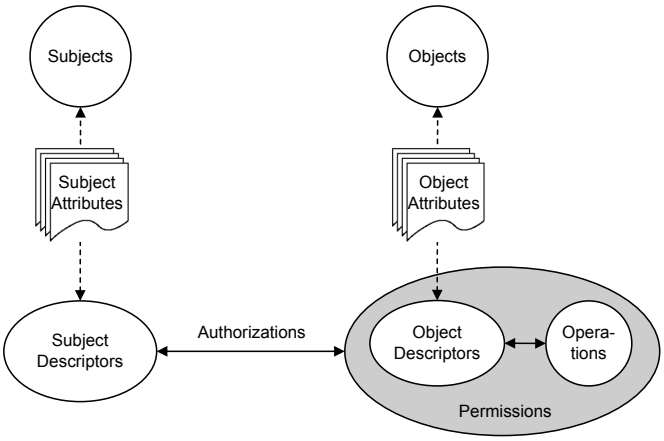

Abbildung 1: Attributbasierte Zugriffskontrolle

Zugriffsrichtlinien [PDMP05].

Im Folgenden stellen wir kurz ein Modell zur attributbasierten Zugriffskontrolle (ABAC) vor, das den genannten Besonderheiten gerecht wird. Eine erste, ausführlichere Beschreibung des Modells ist in [PFMP04] enthalten.

Die Grundidee attributbasierter Zugriffskontrolle ist in Abbildung 1 zu sehen. Sie besteht darin, Zugriffsrechte zwischen den Subjekten und Objekten nicht statisch zu definieren, sondern ihre Eigenschaften oder Attribute dynamisch als Grundlage der Autorisierung zu nutzen. Die Attribute von Subjekten können eher statischer Natur sein, wie z.B. die Position eines Benutzers in einem Unternehmen; sie können jedoch auch dynamischer Natur sein, wie z.B. der aktuelle Aufenthaltsort, das Alter oder auch ein erworbenes Abonnement für eine digitale Bibliothek. Auf Seiten der Objekte werden Metadaten verwendet wie z.B. der Typ oder das Thema einer Ressource. Subjekte und Objekte werden jeweils von einer Menge an Attributen sowie den zugehörigen Werten repräsentiert. Permissions, d.h. Privilegien, bestehen aus der Kombination eines sog. Objektdeskriptors und einer Operation (z.B. Lesen oder Schreiben), die auf den durch den Objektdeskriptor bezeichneten Objekten durchgeführt werden soll. Deskriptoren bestehen allgemein aus einer Menge von Attributen, die jeweils Bedingungen unterworfen werden wie z.B. ’Alter > 17’ oder ’Abonnent = ja’. Eine Rechtezuweisung (Authorization) wird dann zwischen einem Subjektdeskriptor und einem Privileg vorgenommen. Mit Hilfe von Deskriptoren ist es möglich, die im System existierenden Objekte bzw. die anfragenden Subjekte dynamisch den für sie passenden Rechtezuweisungen zuzuordnen und so eine manuelle Rechtevergabe entbehrlich zu machen.

Das ABAC-Modell ist mächtig genug, um die traditionellen Zugriffskontrollmodelle wie benutzerbestimmbare Zugriffskontrolle (DAC), systembestimmte Zugriffskontrolle (MAC) und RBAC abbilden zu können. Genaueres findet sich in [PDMP05].

3 Grafische Modellierung von Zugriffsrichtlinien

In diesem Abschnitt wird eine grafische Methode zum Entwurf von Zugriffsrichtlinien für attributbasierte Modelle entwickelt. Dazu werden in einem ersten Schritt einige Anforderungen, die eine solche Methode aus unserer Sicht erfüllen sollte, aufgestellt:

- Es sollte eine etablierte grafische Notation verwendet werden, so dass die Spezifikationsmethode in die in der Industrie gängigen Entwicklungsprozesse integriert werden kann.
- Ausreichende Werkzeugunterstützung in Form dedizierter Tools oder eine Integration mit verbreiteten Werkzeugen sollte möglich sein; auch im Hinblick auf die spätere automatisierte Transformation der grafischen Spezifikation in eine Repräsentation, die von einem PDP verwendet werden kann.
- Die der grafischen Notation zugrunde liegende Modellierungsprache sollte leichte Erweiterbarkeit aufweisen, so dass keine umfangreiche Neuentwicklung erfolgen muss und die vorhandenen Modellierungswerkzeuge weiter benutzt werden können.

Die Unified Modeling Language (UML) scheint geeignet, die beschriebenen Anforderungen zu erfüllen. Als de-facto Standard für die Modellierung von Softwaresystemen ist sie weitverbreitet und wohlverstanden. Zudem bietet sie einige Ansatzpunkte unterschiedlicher Mächtigkeit zur Erweiterung mit eigenen Sprachkonstrukten. Neben der Möglichkeit, ein vollständig neues Metamodell zu definieren, kann auch das UML-Metamodell unkontrolliert bzw. kontrolliert erweitert werden [HKKR05, S. 334]. Um die bestehende Semantik zu erhalten, verwenden wir die Variante der kontrollierten Erweiterung in Form eines UML-Profils. Da diese Variante auch von den meisten UML-Modellierungstools implementiert werden kann, ist damit die Grundlage für eine umfassende Werkzeugunterstützung gelegt.

3.1 ABAC-Richtlinien mit UML

Dieser Abschnitt stellt zunächst ein konzeptuelles Modell der ABAC-Richtlinien vor. Danach wird eine dedizierte UML-Notation für ABAC-Richtlinien in Form eines Profils erläutert.

Konzeptuelles Muster. Abbildung 2 zeigt ein Muster zum grundlegenden Verständnis der Elemente der attributbasierten Richtlinienspezikation. Es basiert in leicht abgewandelter Form auf den Entwicklungen in [PDMP05], ist jedoch für die Modellierung von Richtlinien angepaßt. So erfolgt im Muster z.B. keine Modellierung von konkreten Benutzern und Ressourcen des Anwendungssystems, da das ABAC-Modell ja gerade von einzelnen Identitäten abstrahiert.

Die Beschreibung der Klassen wurde großenteils bereits analog in Abschnitt 2 vorgenommen.

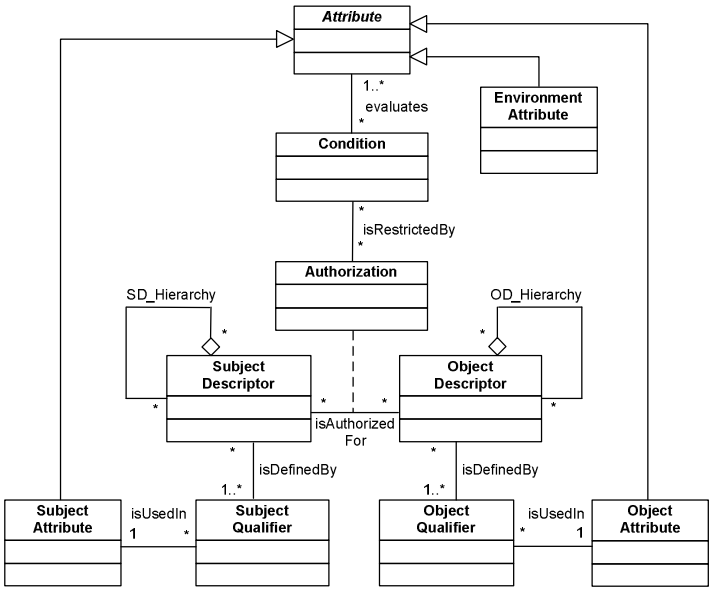

Abbildung 2: Konzeptuelles ABAC-Muster

Neu ist die Möglichkeit, Autorisierungen um zusätzliche Bedingungen zu erweitern. In diesen Bedingungen können zum Einen Subjektattribute mit Objektattributen verglichen werden. Dies ist mit den Deskriptoren alleine nicht möglich, da dort nur Vergleiche auf Basis von konstanten Werten geführt werden. Zum Anderen können Umgebungsattribute wie die Systemzeit oder die Systemlast zusätzlich mit in die Zugriffskontrollentscheidung einbezogen werden, wodurch sich z.B. eine Autorisierung auf die üblichen Geschäftszeiten beschränken lässt.

Autorisierungen werden nun zwischen Subjekt- und Objektdeskriptoren modelliert. Außerdem wird das Konzept des Qualifiers (SubjectQualifier und ObjectQualifier) eingeführt, der für ein Attribut verknüpft mit einer Bedingung steht (z.B. “PLZ beginnt mit 93”). Die Qualifier werden dann zu Deskriptoren zusammengefasst. Ein Deskriptor kann demnach mehreren realen Subjekten bzw. Objekten entsprechen. Subjekt- und Objektdeskriptoren stellen so etwas wie Subjekt- und Objektgruppen dar, nur dass die Zuordnung zu diesen Gruppen nicht explizit, sondern implizit durch Attributwerte erfolgt.

Zur Vereinfachung der grafischen Richtlinien-Spezifikation wurde zusätzlich eine Hierarchisierung der Deskriptoren eingeführt. Dadurch können allgemeine Deskriptoren definiert werden, von denen dann sich nur in einzelnen Qualifiern unterscheidende Spezialisierungen abgeleitet werden können (siehe das Beispiel in Unterabschnitt 3.2).

Die eigentlichen Modellierungselemente für die Zugriffsrichtlinien sind im nun folgenden Profil zusammengefasst; das dargestellte konzeptuelle Muster ist nicht im Sinne eines Metamodells zu verwenden.

UML-Profil. Wie bereits erwähnt, bietet die UML mit den Profilen eine leichtgewichtige

Möglichkeit, spezialisierte Modellierungskonstrukte für bestimmte Anwendungsdomänen bereitzustellen. Da dabei das Metamodell der UML selbst nicht verändert wird, kann die bestehende Syntax und Semantik der UML weiterhin als Grundlage genutzt werden.

Ein UML-Profil ist ein Paket, das als Teil der UML-Infrastruktur definiert ist und dessen Inhalte folglich wie vom ursprünglichen Standard vorgegebene Sprachkonzepte genutzt werden können [HKKR05, S. 335]. Drei Bestandteile können unterschieden werden:

- Stereotypen sind Modellelemente, durch die UML-Metaklassen mit Metaattributen (Tags) und Einschränkungen spezialisiert werden können. Sie sind mit einem Namen versehen und können in einem UML-Diagramm durch textuelle Markierung oder mit einem eigenen Symbol eingesetzt werden.
- Beschränkungen können Vor- oder Nachbedingungen sowie Invarianten für einen bestimmten Stereotyp angeben. Oftmals werden sie in der logikbasierten Object Constraint Language (OCL) angegeben; es ist jedoch auch die Verwendung von natürlicher Sprache oder jeder anderen Programmiersprache möglich.
- Schlüsselwort/Wert-Paare (Tagged Values) sind benannte und typisierte Metaattribute, die einem Stereotyp zugewiesen werden und beliebige Information aufnehmen können.

Wir verwenden Stereotypen hauptsächlich in ihrer einfachsten Form als Klassifizierungsmechanismus für Metaklassen [HKKR05, S. 336]. Die Spezifikation der im Profil enthaltenen Stereotypen findet sich in Tabelle 1. Dort ist neben dem Namen des Stereotyps das jeweils zugehörige Symbol enthalten sowie eine textuelle Beschreibung, die die weitere Definition der Stereotypen vornimmt.

Ergänzend zur Spezifikation der Stereotypen aus Tabelle 1 ist in Tabelle 2 die Verbindung der im ABAC-Muster enthaltenen Klassen zu den definierten Stereotypen gemeinsam mit deren UML-Basisklassen zu sehen. Eine Entsprechung für die Klasse Attribute aus dem Muster ist nicht notwendig, da die Klasse abstrakt ist und in konkreten Richtlinien damit nicht verwendet wird. Die im Muster enthaltene Subjekt- bzw. Objektdeskriptorhierarchisierung (SD_Hierarchy, OD_Hierarchy) wird durch im Richtliniendiagramm durch eine Generalisierung zwischen den jeweiligen Deskriptoren modelliert.

Um ABAC-konforme Klassendiagramme für die Zugriffsrichtlinien zu erreichen, geben wir einige natürlichsprachliche Beschränkungen an; diese sollen die korrekte Benutzung der neuen Elemente sicherstellen:

- Die Stereotypen sind exklusiv auf die Modellelemente anzuwenden, d.h. ein Modellelement darf nur durch einen Stereotyp des Profils stereotypisiert werden.
- Die im Schlüsselwort “Predicate” einer Condition-Assoziationsklasse enthaltene Bedingung darf sich nur auf Attribute beziehen, die mit dieser Klasse assoziiert sind.
- Die einzelnen für die Qualifier angegebenen Einschränkungen müssen sich jeweils auf das assoziierte Attribut beziehen.

Name	Symbol	Beschreibung
SubjectAttribute	SA	Subjektattribute repräsentieren einzelne Eigenschaften von Subjekten des Systems.
ObjectAttribute	OA	Objektattribute repräsentieren einzelne Eigenschaften von Objekten des Systems.
SubjectQualifier	SQ	Subjektqualifier verkörpern einzelne Eigenschaften von Subjekten, die durch einen binären Operator-Operand-Vergleich auf bestimmte Bereiche eingeschränkt wurden. Die Vergleiche werden durch Beschränkungen formuliert, die den Assoziationen zwischen Subjektattributen und Subjektqualifiern zugeordnet werden.
ObjectQualifier	OQ	Objektqualifier verkörpern einzelne Eigenschaften von Objekten, die durch einen binären Operator-Operand-Vergleich auf bestimmte Bereiche eingeschränkt wurden. Die Vergleiche werden durch Beschränkungen formuliert, die den Assoziationen zwischen Objektattributen und Oubjektqualifiern zugeordnet werden.
SubjectDescriptor	SD	Subjektdeskriptoren fassen eine Teilmenge der in Subjektqualifiern enthaltenen eingeschränkten Eigenschaften zusammen und stehen dadurch für Subjekte, die ein bestimmtes Bündel von Eigenschaften aufweisen.
ObjectDescriptor	OD	Objektdeskriptoren fassen eine Teilmenge der in Objektqualifiern enthaltenen eingeschränkten Eigenschaften zusammen und stehen dadurch für Objekte, die ein bestimmtes Bündel von Eigenschaften aufweisen.
Authorization	A	Eine Autorisierung stellt durch eine gerichtete Assoziation dar, dass in einem Subjektdeskriptor beschriebene Subjekte eine bestimmte Operation auf den von einem Objektdeskriptor beschriebenen Objekten ausführen dürfen. Die Assoziation trägt den Namen der Operation.
EnvironmentAttribute	EA	Umgebungsattribute repräsentieren einzelne Systemeigenschaften.
Condition	C	Bedingungen sind zusätzliche Einschränkungen für Autorisierungen. Sie besitzen ein Metaattribut (Tagged Value) “Predicate” vom Typ “string”. Bedingungen können sich auf alle Attributarten beziehen. Sie werden als Assoziationsklasse einer Autorisierung zugeordnet.

Tabelle 1: Stereotypen

ABAC-Modellelement	UML-Metaklasse und Stereotyp	
SubjectAttribute	Class	≪SubjectAttribute≫
ObjectAttribute	Class	≪ObjectAttribute≫
SubjectQualifier	Class	≪SubjectQualifier≫
ObjectQualifier	Class	≪ObjectQualifier≫
SubjectDescriptor	Class	≪SubjectDescriptor≫
ObjectDescriptor	Class	≪ObjectDescriptor≫
Authorization	Association	≪Authorization≫
EnvironmentAttribute	Class	≪EnvironmentAttribute≫
Condition	Class	≪Condition≫

Tabelle 2: Mapping zwischen konzeptuellem ABAC-Modell und UML-Profil

- Die im Muster in Abb. 2 bei den Assoziationen angegebenen Kardinalitäten sind in den Richtliniendiagrammen zu berücksichtigen.

3.2 Ein Beispielszenario

Ein exemplarischer Anwendungsfall soll den Einsatz des Profils verdeutlichen. Eine Wirtschaftsauskunftei mit entsprechenden Befugnissen (z.B. die Schufa) betreibt im Internet einen Dienst in Form eines Web Services, der Informationen über Merkmale und Scorewerte von Personen anbietet. Benutzer können Kreditinstitute sowie natürliche Personen sein. Kreditinstitute sollen im Rahmen von Bonitätsprüfungen Merkmale beliebiger Personen abfragen können. Desweiteren sollen durch die beteiligten Institute als Partner der Auskunftei Merkmale von Personen hinzugefügt bzw. aktualisiert (verändert) werden können. Außerdem sind Personen berechtigt, bei der Auskunftei die sie betreffenden Merkmale abzufragen. Kreditinstitute, die besondere Verträge (Goldstatus) mit der Auskunftei haben, können zudem eine Kundenbewertung in Form eines Scorewerts anfordern. Die angegebenen Beschränkungen sollen zudem nur für Personen und Institute aus Deutschland angewendet werden. Abbildung 3 zeigt den grafischen Entwurf demgemäßer Richtlinien für das vorliegende Szenario.

Wie aus der Abbildung ersichtlich, werden die stereotypisierten Klassen der Übersichtlichkeit halber in einer verkürzten Notation ohne Attribute bzw. Operationen gezeichnet.

Insgesamt sind drei Subjektattribute (Ort, Typ, Status) modelliert, die über mit Einschränkungen versehenen Assoziationen den Anforderungen entsprechende Qualifier definieren. Die Qualifier wiederum werden so mit Subjektdeskriptor-Klassen assoziiert, dass die gewünschten Eigenschaften der Web-Service-Benutzer zusammengefasst werden. Im Einzelnen resultieren ein Deskriptor, der für Personen aus Deutschland steht (Person_DE), sowie zwei Deskriptoren, die für deutsche Kreditinstitute mit bzw. ohne Goldstatus stehen. Zu beachten ist hier die Verwendung der Hierarchisierung der Deskriptoren: Der Deskriptor für Kreditinstitute mit Goldstatus ist vom allgemeinen Kreditinstitut-Deskriptor abgeleitet, so dass nur noch eine Assoziation mit dem Hat_Goldstatus-Qualifier nötig ist.

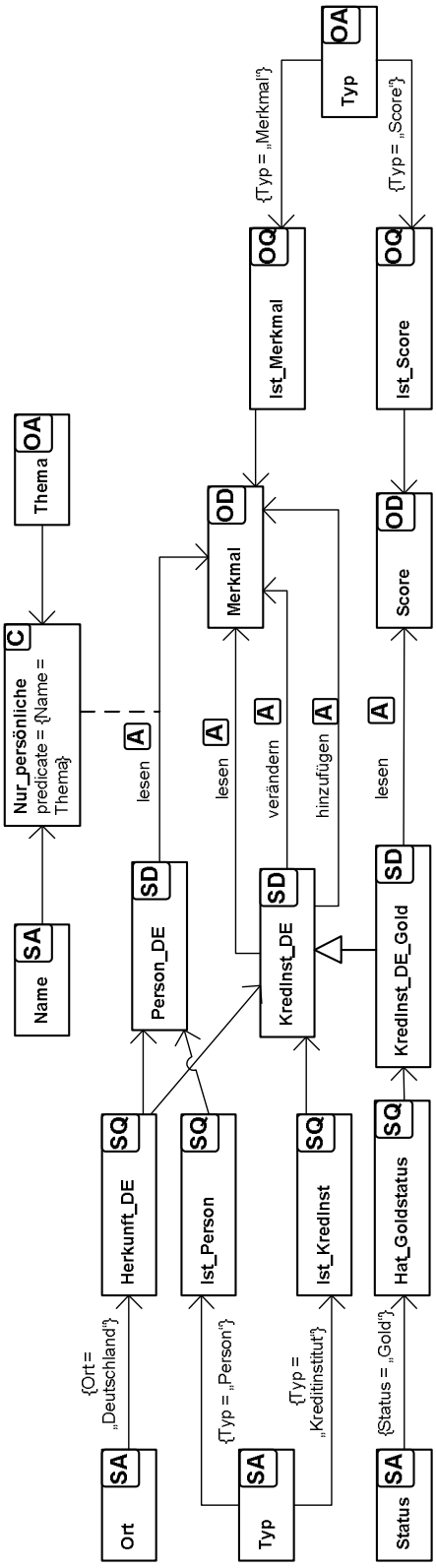

Abbildung 3: Zugriffsrichtlinien für die Auskunftei

Ähnliches zeigt sich auf der Objektseite; hier finden sich zwei Objektdeskriptoren für Personenmerkmale bzw. -scores, die sich nur in ihrem Objekttyp unterscheiden. Die Deskriptoren werden über ein entsprechendes Objektattribut und passende Qualifier gebildet.

Die Autorisierungen selbst sind über gerichtete Assoziationen modelliert, die jeweils mit dem erteilten Zugriffsrecht benannt sind.

Die Umsetzung der Anforderung, daß jede Person die eigenen Merkmale lesen können soll, verdeutlicht die Leistungsfähigkeit des ABAC-Modells. Dies wird hier durch eine Condition-Assoziationsklasse erreicht, die als Schlüsselwort/Wert-Paar ein Prädikat enthält, das das zusätzliche Subjektattribut Name mit dem Objektattribut Thema (bezeichnend den Gegenstand eines Objekts, hier also die Person, die ein Merkmal aufweist) vergleicht und mit ihnen assoziiert ist. Das Metaattribut bei den Condition-Klassen wird entgegen der in [HKKR05, S. 340] beschriebenen Notation aus Gründen der Übersichtlichkeit im obersten Abschnitt des Klassensymbols hinterlegt. Die Ausführungen in Abschnitt 2 wieder aufgreifend, illustriert diese Anforderung die Grenzen des rollenbasierten Zugriffskontrollmodells, das für jede Person eine eigene Rolle bereitstellen müsste.

4 Verwandte Arbeiten

Die grafische Modellierung von Sicherheitsaspekten von Anwendungssystemen allgemein und Zugriffskontrolle im speziellen ist ein aktuelles und vielbehandeltes Thema. Der nachfolgende Überblick beschränkt sich auf Arbeiten, die ebenfalls die UML zur Modellierung verwenden.

Eine Methode zum grafischen Entwurf von sicheren Data Warehouses bietet [VFMP06]; sie bietet ein UML-Profil sowie eine Erweiterung von OCL, um Sicherheitsbeschränkungen wie z.B. Autorisierungsinformationen in Klassendiagrammen von multidimensionalen Data Warehouses repräsentieren zu können. Dieser Ansatz ist für unsere Zwecke ungeeignet, da keine Separation von Daten und Zugriffsbeschränkungen erfolgt, so dass die Sicherheitsinformationen nicht in einem Web-Service-Umfeld verwendet werden können.

Jürjens präsentiert in [Jür02] ein umfassendes UML-Profil namens UMLsec zur Entwicklung sicherer Systeme. Dieses Profil kann Anforderungen u.a. der Vertraulichkeit, des Informationsflusses und der sicheren Kommunikation in verteilten Systemen abbilden. Darüber hinausgehend wird für eine Untermenge der UML eine formale Semantik angegeben, mit der eine Evaluation der Modellbildung durchgeführt werden kann. Jedoch können keine dedizierten Zugriffsrichtlinien modelliert werden.

SecureUML von Basin et al. [BMPP06] ist ein modellgetriebener Ansatz zur Entwicklung von sicheren Systemen. Dabei werden die in der Systemspezifikation angegeben Sicherheitsanforderungen zu Programmcode einer Zugriffskontrollinfrastruktur umgesetzt. Dies geschieht auf der Basis eines erweiterten RBAC-Modells, das zwar zusätzliche Autorisierungseinschränkungen in OCL formulieren kann - ähnlich der Conditions in ABAC -, jedoch nicht die volle Ausdruckskraft sowie Loslösung von Identitäten, wie sie für offene Systeme notwendig ist, erreicht.

Ebenfalls im RBAC-Umfeld bewegt sich [RLFK04], die ein UML-Referenzmuster für die Anwendungsentwicklung mit RBAC angeben sowie beschreiben, wie Beschränkungen zur Aufgabentrennung grafisch modelliert und validiert werden können. Eine explizite Richtlinienspezifikation erfolgt nicht.

In [SAGM05] wird ein Ansatz vorgestellt, der unter Verwendung eines Tools die Elemente des RBAC-Modells nachbildet sowie weitere RBAC-spezifische Beschränkungen in OCL beschreibt. Die so dargestellten Richtlinien können mit dem genannten Tool validiert werden.

Der Vorteil des vorliegenden Ansatzes gegenüber den in diesem Abschnitt vorgestellten Arbeiten liegt zusammengefaßt darin, dass nur mit unserem Profil attributbasierte Richtlinien modelliert werden können, die für offene System geeignet sind.

5 Zusammenfassung und Ausblick

In diesem Beitrag wurde eine Erweiterung der UML in Form eines Profils präsentiert, das zur Modellierung von attributbasierten Zugriffsrichtlinien dienen kann. Vorangehend wurde motiviert, warum im Umfeld von Web Services bzw. allgemeiner offener Systeme ein solches Zugriffskontrollmodell notwendig ist, warum eine separate, von den Daten losgelöste Spezifikation der Zugriffsrichtlinien sinnvoll ist und weshalb eine weitverbreitete grafische Notation wie die UML die geeignete Grundlage für ein solche Spezifikation sein kann. Abrundend wurde der Einsatz der entwickelten Notation an einem Beispiel erläutert. Insgesamt erlaubt der dargestellte Ansatz, Zugriffsrichtlinien einheitlich und durch die grafische Darstellung auch dem nicht sehr geübten Administrator zugänglich zu spezifizieren.

Für zukünftige Arbeiten bieten sich mehrere Richtungen an. Ein wichtiger erster Schritt besteht in der Implementierung des Tool-Supports für das Management der Zugriffsrichtlinien. Die vorgestellte Notation soll dafür als Grundlage dienen. Dies schließt eine Funktionalität mit ein, die die spezifizierten Richtlinien ausgehend von XMI, dem XML Metadata Interchange der UML, auf Grundlage eines Transformationsschemas weitestgehend automatisiert in Sprachen wie RuleML oder XACML übersetzt. Diese XML-Daten können danach von einem Zugriffskontrollmodul (das die Funktionen von PDP und PEP übernimmt), wie es z.B. in [BMPP06] vorgestellt wurde, verarbeitet werden. Daneben streben wir die Definition von Funktionalitäten für die Administration von ABAC-Richtlinien an, wie es in ähnlicher Weise in [DMP04] für RBAC beschrieben wurde. Schließlich wollen wir eine Praxiserprobung und Evaluation durchführen. Dies soll im Umfeld des EU-Projekts Access-eGov[3] geschehen, das sich die Entwicklung einer semantischen Web-Service-Infrastruktur mit Peer-to-Peer-Elementen für das eGovernment zum Ziel gesetzt hat. Es ist beabsichtigt, die vorliegenden Ansätze bei der Entwicklung der Sicherheitsinfrastruktur zu verwenden.

[3] `http://www.access-egov.org`

Literatur

[BMPP06] David Basin, Jürgen Doser und Torsten Lodderstedt. Model Driven Security: From UML Models to Access Control Infrastructures. *ACM Transactions on Software Engineering and Methodology*, 15(1):39-91, 2006.

[BMPP06] Sönke Busch, Björn Muschall, Günther Pernul und Torsten Priebe. Authrule: A Generic Rule-Based Authorization Module. In *Proceedings of the 20th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, Seiten 267-281, 2006.

[DDCdVS05] E. Damiani, S. De Capitani di Vimercati und P. Samarati. New Paradigms for Access Control in Open Environments. In *Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology (ISSPIT 2005)*, Seiten 540-545, 2005.

[DMP04] Fredj Dridi, Björn Muschall und Günther Pernul. Administration of an RBAC System. In *Proceedings of the 37th Hawaii International Conference on System Sciences (HICSS-37)*, 2004

[FSG+01] David F. Ferraiolo, Ravi S. Sandhu, Serban I. Gavrila, D. Richard Kuhn und Ramaswamy Chandramouli. Proposed NIST Standard for Role-based Access Control. *ACM Transactions on Information and System Security*, 4(3):224-274, 2001.

[HKKR05] Martin Hitz, Gerti Kappel, Elisabeth Kapsammer und Werner Retschitzegger. *UML@Work*. dpunkt, 3. Auflage, 2005.

[Jür02] Jan Jürjens. UMLsec: Extending UML for Secure Systems Development. In *Proceedings of the 5th International Conference on the Unified Modeling Language (UML 2002)*, Seiten 412-425, 2002.

[KMPP05] Manuel Koch, Luigi V. Mancini und Francesco Parisi-Presicce. Graph-based specification of access control policies. *Journal of Computer and System Sciences*, 71(1):1-33, 2005.

[OAS05] OASIS. eXtensible Access Control Markup Language (XACML) Version 2.0, 2005.

[PDMP05] Torsten Priebe, Wolfgang Dobmeier, Björn Muschall und Günther Pernul. ABAC - Ein Referenzmodell für attributbasierte Zugriffskontrolle. In *Tagungsband der 2. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik (Sicherheit 2005)*, Seiten 285-296, 2005.

[PFMP04] Torsten Priebe, Eduardo B. Fernández, Jens Ingo Mehlau und Günther Pernul. A Pattern System for Access Control. In *Proceedings of the 18th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, Seiten 235-249, 2004.

[RLFK04] Indrakshi Ray, Na Li, Robert B. France und Dae-Kyoo Kim. Using UML to Visualize Role-based Access Control Constraints. In *Proceedings of the 9th ACM Symposium on Access Control Models and Technologies (SACMAT 2004)*, Seiten 115-124, 2004.

[SAGM05] Karsten Sohr, Gail-Joon Ahn, Martin Gogolla und Lars Migge. Specification and Validation of Authorisation Constraints Using UML and OCL. In *Proceedings of the 10th European Symposium on Research in Computer Security (ESORICS 2005)*, Seiten 64-79, 2005.

[VFMP06] Rodolfo Villarroel, Eduardo Fernández-Medina und Mario Piattini. A UML 2.0/OCL Extension for Designing Secure Data Warehouses. *Journal of Research and Practice in Information Technology*, 38(1):31-43, 2006.

[YT05] Eric Yuan und Jin Tong. Attribute Based Access Control (ABAC) for Web Services. In *Proceedings of the 2005 IEEE International Conference on Web Services (ICWS 2005)*, Seiten 561-569, 2005.

[ZRG04] Nan Zhang, Mark Ryan und Dimitar P. Guelev. Synthesising Verified Access Control Systems in XACML. In *Proceedings of the 2004 ACM Workshop on Formal Methods in Security Engineering (FMSE 2004)*, Seiten 56-65, 2004